2FA + Session Management

Real-World Integration

🔄 Traditional vs 2FA Login Flow

Traditional Login:

```
Username/Password → ✅ → Full Session → Dashboard
```

2FA Login:

```
Username/Password → ✅ → Pending Session → TOTP Code → ✅ → Full Session → Dashboard
```

Key difference: Two-step verification with intermediate session state

🎯 Session States in 2FA

State	Description	What User Can Do
`unauthenticated`	No login attempt	Login page only
`pending_2fa`	Password verified	Enter TOTP code
`authenticated`	Full login complete	Access all features

💻 Implementation: Session-Based 2FA

File: `SessionAuth.php`

```php
<?php
session_start();

class SessionAuth {

    // Check if user is fully authenticated
    public static function isAuthenticated() {
        return isset($_SESSION['user_id']) &&
               $_SESSION['auth_status'] === 'authenticated';
    }

    // Check if user is pending 2FA
    public static function isPending2FA() {
        return isset($_SESSION['pending_user']) &&
               $_SESSION['auth_status'] === 'pending_2fa';
    }

    // Start login process (after password verification)
    public static function startLogin($username) {
        $_SESSION['pending_user'] = $username;
        $_SESSION['auth_status'] = 'pending_2fa';
        $_SESSION['login_attempts'] = 0;
        $_SESSION['login_time'] = time();
    }

    // Complete login (after 2FA verification)
    public static function completeLogin($user_id) {
        unset($_SESSION['pending_user']);
        $_SESSION['user_id'] = $user_id;
        $_SESSION['auth_status'] = 'authenticated';
        $_SESSION['login_complete_time'] = time();
    }

    // Logout user
    public static function logout() {
        session_destroy();
        session_start();
```

🔐 Step 1: Password Login

File: `login.php`

```php
<?php
require_once 'SessionAuth.php';
require_once 'config.php';

if ($_POST['username'] && $_POST['password']) {
    $username = $_POST['username'];
    $password = $_POST['password'];

    // Verify username/password
    $user = getUser($username);
    if ($user && password_verify($password, $user['password'])) {

        // Check if user has 2FA enabled
        if ($user['totp_enabled']) {
            // Start 2FA process
            SessionAuth::startLogin($username);
            header('Location: verify_2fa.php');
            exit;
        } else {
            // No 2FA - complete login immediately
            SessionAuth::completeLogin($user['id']);
            header('Location: dashboard.php');
            exit;
        }
    } else {
        $error = "Invalid username or password";
    }
}
?>

<!DOCTYPE html>
<html>
<head><title>Login</title></head>
<body>
    <h2>Login</h2>
    <?php if (isset($error)): ?>
        <p style="color: red;"><?= htmlspecialchars($error) ?></p>
    <?php endif; ?>

    <form method="POST">
        <input type="text" name="username" placeholder="Username" required><br>
        <input type="password" name="password" placeholder="Password" required><br>
        <button type="submit">Login</button>
```

📱 Step 2: 2FA Verification

File: `verify_2fa.php`

```php
<?php
require_once 'SessionAuth.php';
require_once 'SimpleTOTP.php';
require_once 'config.php';

// Redirect if not in pending 2FA state
if (!SessionAuth::isPending2FA()) {
    header('Location: login.php');
    exit;
}

// Handle 2FA verification
if ($_POST['totp_code']) {
    $totp_code = $_POST['totp_code'];
    $username = $_SESSION['pending_user'];

    // Get user's TOTP secret
    $user = getUser($username);
    if ($user && $user['totp_enabled']) {
        $secret = base64_decode($user['totp_secret']);

        // Verify TOTP code
        if (SimpleTOTP::verifyCode($totp_code, $secret)) {
            // Success! Complete login
            SessionAuth::completeLogin($user['id']);
            header('Location: dashboard.php');
            exit;
        } else {
            // Track failed attempts
            $_SESSION['login_attempts']++;
            if ($_SESSION['login_attempts'] >= 3) {
                SessionAuth::logout();
                header('Location: login.php?error=too_many_attempts');
                exit;
            }
            $error = "Invalid 2FA code. Attempts: " . $_SESSION['login_attempts'] . "/3";
        }
    }
}

// Check for timeout (5 minutes)
if (time() - $_SESSION['login_time'] > 300) {
    SessionAuth::logout();
    header('Location: login.php?error=timeout');
    exit;
}
?>

<!DOCTYPE html>
<html>
<head><title>Two-Factor Authentication</title></head>
<body>
    <h2>Enter 2FA Code</h2>
    <p>Please enter the 6-digit code from your authenticator app</p>

    <?php if (isset($error)): ?>
        <p style="color: red;"><?= htmlspecialchars($error) ?></p>
    <?php endif; ?>

    <form method="POST">
        <input type="text" name="totp_code" placeholder="123456" maxlength="6" required
               pattern="[0-9]{6}" autofocus><br>
        <button type="submit">Verify</button>
        <a href="login.php">Cancel</a>
    </form>

    <p><small>Code expires in: <span id="countdown"></span></small></p>

    <script>
    // Show countdown timer for current TOTP window
    function updateCountdown() {
        const now = Math.floor(Date.now() / 1000);
        const timeLeft = 30 - (now % 30);
        document.getElementById('countdown').textContent = timeLeft + 's';
    }
```

🏠 Step 3: Protected Dashboard

File: `dashboard.php`

```php
<?php
require_once 'SessionAuth.php';
require_once 'config.php';

// Require full authentication
if (!SessionAuth::isAuthenticated()) {
    header('Location: login.php');
    exit;
}

$user = getUser($_SESSION['user_id']);
?>

<!DOCTYPE html>
<html>
<head><title>Dashboard</title></head>
<body>
    <h2>Welcome, <?= htmlspecialchars($user['username']) ?>!</h2>

    <div class="session-info">
        <h3>Session Information:</h3>
        <p>User ID: <?= $_SESSION['user_id'] ?></p>
        <p>Login Time: <?= date('Y-m-d H:i:s', $_SESSION['login_complete_time']) ?></p>
        <p>2FA Status: <?= $user['totp_enabled'] ? 'Enabled' : 'Disabled' ?></p>
    </div>

    <div class="actions">
        <a href="profile.php">Edit Profile</a> |
        <a href="2fa_setup.php">Manage 2FA</a> |
        <a href="logout.php">Logout</a>
    </div>
```

🚪 Logout and Session Management

File: `logout.php`

```php
<?php
require_once 'SessionAuth.php';

// Log the logout for security monitoring
if (SessionAuth::isAuthenticated()) {
    $logout_time = date('Y-m-d H:i:s');
    // You could log this to database for audit trail
    error_log("User {$_SESSION['user_id']} logged out at {$logout_time}");
}

SessionAuth::logout();
header('Location: login.php?message=logged_out');
?>
```

🛡️ Security Middleware

File: `middleware.php`

```php
<?php
require_once 'SessionAuth.php';

class SecurityMiddleware {

    // Protect pages that require full authentication
    public static function requireAuth() {
        if (!SessionAuth::isAuthenticated()) {
            header('Location: login.php?redirect=' . urlencode($_SERVER['REQUEST_URI']));
            exit;
        }
    }

    // Protect 2FA setup pages
    public static function require2FAState() {
        if (!SessionAuth::isPending2FA()) {
            header('Location: login.php');
            exit;
        }
    }

    // Prevent authenticated users from accessing login pages
    public static function redirectIfAuthenticated() {
        if (SessionAuth::isAuthenticated()) {
            header('Location: dashboard.php');
            exit;
        }
    }

    // Session timeout check
    public static function checkTimeout($max_idle_time = 1800) { // 30 minutes
        if (isset($_SESSION['last_activity'])) {
            if (time() - $_SESSION['last_activity'] > $max_idle_time) {
                SessionAuth::logout();
                header('Location: login.php?error=session_timeout');
                exit;
            }
        }
        $_SESSION['last_activity'] = time();
    }
}
```

🔄 Complete Session Flow Example

```php
<?php
// In any protected page
require_once 'middleware.php';
SecurityMiddleware::requireAuth(); // Ensures full authentication

// In login page
SecurityMiddleware::redirectIfAuthenticated(); // Prevents double login

// In 2FA verification page
SecurityMiddleware::require2FAState(); // Ensures proper flow

// Session state transitions:
// null → pending_2fa → authenticated → null (logout)
?>
```

🎯 Key Benefits of Session Integration

Security:

- Prevents bypassing 2FA
- Time-limited authentication attempts
- Proper session invalidation

User Experience:

- Seamless flow between authentication steps
- Maintains state during process
- Handles errors gracefully

System Design:

- Clear separation of authentication states

🧪 Testing the Integration

```php
// Test script: test_session_flow.php
session_start();

echo "Current session state:\n";
echo "- Auth status: " . ($_SESSION['auth_status'] ?? 'none') . "\n";
echo "- User ID: " . ($_SESSION['user_id'] ?? 'none') . "\n";
echo "- Pending user: " . ($_SESSION['pending_user'] ?? 'none') . "\n";
echo "- Is authenticated: " . (SessionAuth::isAuthenticated() ? 'Yes' : 'No') . "\n";
echo "- Is pending 2FA:
```